Méthodes numériques

Chapitre 1 :

Généralités sur l'analyse numérique et le calcul scientifique

1.1 Motivations.

Un algorithme est un énoncé décrivant, à l'aide d'opérations élémentaires, toutes les étapes d'une démarche systématique permettant la résolution d'un problème spécifique. Un algorithme peut à son tour contenir des sous-algorithmes et doit pouvoir s'achever après un nombre fini d'opérations élémentaires afin de pouvoir être utilisé dans un programme informatique. La mise en œuvre d'un algorithme consiste en l'écriture de la série d'opérations élémentaires le composant dans un langage de programmation, ce que l'on appelle aussi fréquemment une implémentation.

La complexité d'un algorithme est une mesure de son temps d'exécution. Calculer la complexité d'un algorithme fait donc partie de l'analyse de l'efficacité et du coût d'une méthode numérique. pseudo-langage pour la description des algorithmes

on compte les soustractions comme des additions.

1.2 Représentation d'un nombre en machine, erreurs d'arrondis.

- **Un exemple : calcul approché de π.**

Nous savons aujourd'hui que l'aire d'un cercle de rayon r est $A=\pi r^2$. Parmi les solutions proposées pour approcher A, une méthode consiste à construire un polygône dont le nombre de côté augmenterai jusqu'à ce qu'il devienne équivalent au cercle circonscrit. C'est Archimède vers 250 avant J-C qui appliquera cette propriété au calcul des décimales du nombre π, en utilisant à la fois un polygône inscrit et circonscrit au cercle. Il utilise ainsi un algorithme pour le calcul et parvient à l'approximation de π dans l'intervalle (3+1/7,3+10/71) en faisant tendre le nombre de côtés jusqu'à 96.

On considère un cercle de rayon r=1 et on note A_n l'aire associée au polygône inscrit à n côtés.

.

En notant $\alpha_n=2\pi/n$, A_n est égale à n fois l'aire du triangle ABC représenté sur la figure 1.1, c'est-à-dire.

$$A_n = n\cos\frac{\alpha_n}{2}\sin\frac{\alpha_n}{2}$$

$$A_n = \frac{n}{2}(2\cos\frac{\alpha_n}{2}\sin\frac{\alpha_n}{2}) = \frac{n}{2}\sin\alpha_n = \frac{n}{2}\sin(\frac{2\pi}{n})$$

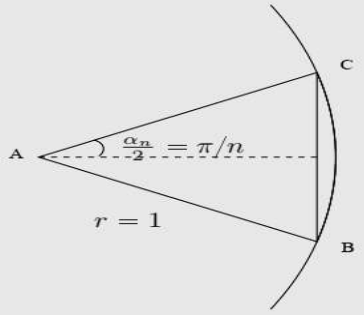

FIG. 1.1 – Quadrature du cercle

Comme on cherche à calculer π à l'aide de A_n, on ne peut pas utiliser l'expression ci-dessus pour calculer An, mais on peut exprimer A_{2n} en fonction de
An en utilisant la relation.

$$\sin\frac{\alpha_n}{2} = \sqrt{\frac{1-\cos\alpha_n}{2}} = \sqrt{\frac{1-\sqrt{1-\sin^2\alpha_n}}{2}}.$$

Ainsi, en prenant $n = 2^k$, on définit l'approximation de π par récurrence.

$$x_k = A_{2^k} = \frac{2^k}{2}s_k, \quad \text{avec } s_k = \sin(\frac{2\pi}{2^k})$$

En partant de k =2 (i.e. n =4 et s =1) on obtient l'algorithme suivant :

Algorithm 1.1 Algorithme de calcul de π, version naïve

1: $s \leftarrow 1$, $n \leftarrow 4$ ▷ Initialisations
2: **Tantque** $s > 1e-10$ **faire** ▷ Arrêt si $s = \sin(\alpha)$ est petit
3: $s \leftarrow sqrt((1 - sqrt(1 - s * s))/2$ ▷ nouvelle valeur de $\sin(\alpha/2)$
4: $n \leftarrow 2 * n$ ▷ nouvelle valeur de n
5: $A \leftarrow (n/2) * s$ ▷ nouvelle valeur de l'aire du polygône
6: **fin Tantque**

On a $\lim_{k\longrightarrow+\infty} x_k = \pi$ Ce n'est pourtant pas du tout ce que l'on va observer sur machine ! Les résultats en Matlab de la table 1.1 montre que l'algorithme

commence pas converger vers π puis pour $n \geq 65536$, l'erreur augmente et finalement on obtient An=0!! "Although the theory and the program are correct.
Ce ci résulte du codage des valeurs réelles sur un nombre fini de bits.

n	A_n	$A_n - \pi$	$\sin(\alpha_n)$
4	2.000000000000000	-1.141592653589793	1.000000000000000
8	2.828427124746190	-0.313165528843603	0.707106781186548
16	3.061467458920719	-0.080125194669074	0.382683432365090
32	3.121445152258053	-0.020147501331740	0.195090322016128
64	3.136548490545941	-0.005044163043852	0.098017140329561
128	3.140331156954739	-0.001261496635054	0.049067674327418
256	3.141277250932757	-0.000315402657036	0.024541228522912
512	3.141513801144145	-0.000078852445648	0.012271538285719
1024	3.141572940367883	-0.000019713221910	0.006135884649156
2048	3.141587725279961	-0.000004928309832	0.003067956762969
4096	3.141591421504635	-0.000001232085158	0.001533980186282
8192	3.141592345611077	-0.000000307978716	0.000766990318753
16384	3.141592576545004	-0.000000077044789	0.000383495187567
32768	3.141592633463248	-0.000000020126545	0.000191747597257
65536	3.141592654807589	0.000000001217796	0.000095873799280
131072	3.141592645321215	-0.000000008268578	0.000047936899495
262144	3.141592607375720	-0.000000046214073	0.000023968449458
524288	3.141592910939673	0.000000257349880	0.000011984225887
1048576	3.141594125195191	0.000001471605398	0.000005992115260
2097152	3.141596553704820	0.000003900115026	0.000002996059946
4194304	3.141596553704820	0.000003900115026	0.000001498029973
8388608	3.141674265021758	0.000081611431964	0.000000749033514
16777216	3.141829681889202	0.000237028299408	0.000000374535284
33554432	3.142451272494134	0.000858618904341	0.000000187304692
67108864	3.142451272494134	0.000858618904341	0.000000093652346
134217728	3.162277660168380	0.020685006578586	0.000000047121609
268435456	3.162277660168380	0.020685006578586	0.000000023560805
536870912	3.464101615137754	0.322508961547961	0.000000012904784
1073741824	4.000000000000000	0.858407346410207	0.000000007450581
2.147484e+09	0.000000000000000	-3.141592653589793	0.000000000000000

TAB. 1.1 – Calcul de π avec l'algorithme naïf 1.1

1.3 Représentation scientifique des nombres dans différentes bases

- **Partie entière, mantisse et exposant**
- **Exemple en base 10**

  La base 10 est la base naturelle avec laquelle on travaille et celle que l'on retrouve dans les calculatrices. Un nombre à virgule, ou nombre décimal, à plusieurs écritures différentes en changeant simplement la position du point décimal et en rajoutant à la fin une puissance de 10 dans l'écriture de ce nombre. La partie à gauche du point décimal est la partie entière, celle à droite avant l'exposant s'appelle la mantisse. Par exemple le nombre ***x* =1234.5678** à plusieurs représentations :

  x =1234.5678 =1234.5678 10^0 =1.2345678 10^3 =0.0012345678 10^6

  avec

  – *Partie entière* : 1234

  – *Mantisse* : 0.5678 ou 1.2345678 ou 0.0012345678

  – *Exposant* : 4 ou 6

- **Exemple en base 2**

  C'est la base que les ordinateurs utilisent. Les chiffres utilisables en base 2 sont 0 et 1 que l'on appelle *bit* pour *binary digit*, les ordinateurs travaillent en binaire. Par exemple.

$$39 = 32 + 4 + 2 + 1 = 2^5 + 2^2 + 2^1 + 2^0 = (100111)_2,$$
$$3.625 = 2^1 + 2^0 + 2^{-1} + 2^{-3} = (11.101)_2 = (1.1101)_2\ 2^1$$

1.4 Représentation d'un nombre en machine : nombres flottants

De façon générale tout nombre réels x sera représenté dans une base b (b =10 pour une calculatrice b =2 pour un ordinateur) par son signe (+ ou -), la mantisse m (appelée aussi significande) , la base b et un exposant e tel que le couple (m,e) caractérise le nombre. En faisant varier e, on fait « flotter » la virgule décimale. La limitation fondamentale est que la place mémoire d'un ordinateur est limitée, c'est-à-dire qu'il ne pourra stocker qu'un ensemble fini de nombres. Ainsi un nombre machine réel ou *nombre à virgule flottante* s'écrira

$$\begin{aligned} \tilde{x} &= \pm m \cdot b^e \\ m &= D.D\cdots D \\ e &= D\cdots D \end{aligned}$$

ou D ∈ (0,1, ..., b -1) représente un chiffre.

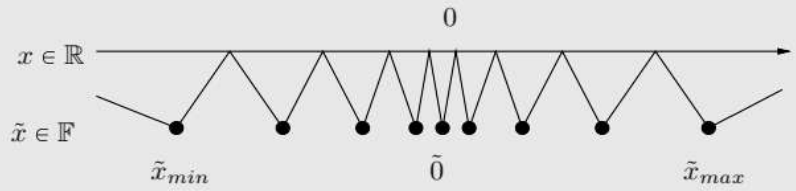

FIG. 1.2 – Représentation des nombres réels $\mathbb{R}$ par les nombres machine $\mathbb{F}$

1.5 Calculs sur les nombres flottants

- **Erreurs d'arrondi**

Si $x\tilde{}$ et $y\tilde{}$ sont deux nombres machine, alors $z = x\tilde{} * y\tilde{}$ ne correspondra pas en général à un nombre machine puisque le produit demande une quantité double de chiffres. Le résultat sera un nombre machine $z\tilde{}$ proche de z.

On définit l'*erreur absolue* entre un nombre réel x et le nombre machine correspondant $x\tilde{}$ par

$$r_a = |x - \tilde{x}|.$$

L'*erreur relative* entre ces nombres (si $x \neq 0$) est définie par

$$r = \frac{|x - \tilde{x}|}{|x|}.$$

Opérations machine : On désigne par *flop* (de l'anglais *floating operation*) une opération élémentaire à virgule flottante (addition, soustraction, multiplication ou division) de l'ordinateur. Sur les calculateurs actuels on peut s'attendre à la précision suivante, obtenue dans les opérations basiques :

$$\tilde{x} \tilde{\oplus} \tilde{y} = (\tilde{x} \oplus \tilde{y})(1 + r)$$

où $|r| < eps$, la précision machine, et $\oplus$ représente l'opération exacte, $\oplus \in \{+, -, *, /\}$ et $\tilde{\oplus}$ représente l'opération de l'ordinateur (*flop*).

Le nombre machine (*eps*) est le plus petit nombre machine tel que 1+ *eps* >1 sur la machine.

- **Erreurs d'annulation**
  Ce sont les erreurs dûes à l'annulation numérique de chiffres significatifs, quand les nombres ne sont représentés qu'avec une quantité finie de chiffres, comme les nombres machine. Il est donc important en pratique d'être attentifs aux signes dans les expressions, comme l'exemple suivant le montre :

Exemple : on cherche à évaluer sur l'ordinateur de façon précise, pour de petites valeurs de x, la fonction

$$f(x) = \frac{1}{1-\sqrt{1-x^2}}.$$

Pour $|x| < \sqrt{eps}$, on risque d'avoir le nombre $\sqrt{1-x^2}$ remplacé par 1, par la machine, et donc lors du calcul de $f(x)$, on risque d'effectuer une division par 0. Par exemple, pour $x = \frac{\sqrt{eps}}{2}$, on obtient :

```
>> f=inline('1./(1-sqrt(1-x.^2))','x');
>> f(0.5*sqrt(eps))

ans =

   Inf
```

On ne peut donc pas évaluer précisément $f(x)$ sur l'ordinateur dans ce cas. Maintenant, si on multiplie dans $f(x)$ le numérateur et le dénominateur par $1+\sqrt{1-x^2}$, on obtient

$$f(x) = \frac{1+\sqrt{1-x^2}}{x^2}$$

Cette fois, on peut évaluer $f(x)$ de façon précise :

```
>> f=inline('(1+sqrt(1-x.^2))./x.^2','x')

>> f(0.5*sqrt(eps))

ans =

   3.6029e+16
```

1.6 Quelques catastrophes dues à l'arithmétique flottante

Il y a un petit nombre "connu" de catastrophes dans la vie réelle qui sont attribuables à une mauvaise gestion de l'arithmétique des ordinateurs (erreurs d'arondis, d'annulation).

Exemple :

*** Missile Patriot(1991) * Explosion d'Ariane 5 (1996) * Bourse de Vancouver(1982)**

Chapitre 2

Méthodes directes de résolution des systèmes linéaires

On considère la résolution du système linéaire

$$\boldsymbol{Ax} = \boldsymbol{b},$$

avec A une matrice d'ordre n à coefficients réels inversible et $\boldsymbol{b}$ un vecteur de R_n, par des méthodes dites directes, c'est-à-dire fournissant, en l'absence d'erreurs d'arrondi, la solution exacte en un nombre fini d'opérations élémentaires. On verra que ces méthodes consistent en la construction d'une matrice inversible M telle que $M\,A$ soit une matrice triangulaire, le système linéaire équivalent (au sens où il possède la même solution) obtenu.

$$MA\boldsymbol{x} = M\boldsymbol{b},$$

étant alors « facile » à résoudre (on verra ce que l'on entend précisément par là). Une telle idée est par exemple à la base de la célèbre méthode d'élimination de Gauss , qui permet de ramener la résolution d'un système linéaire quelconque à celle d'un système triangulaire supérieur.

2.1 Remarques sur la résolution des systèmes triangulaires

Observons tout d'abord que la solution du système linéaire $A\boldsymbol{x} = \boldsymbol{b}$, avec A une matrice inversible, ne s'obtient pas en inversant A, puis en calculant le vecteur $A^{-1}\boldsymbol{b}$, mais en réalisant plutôt des combinaisons linéaires sur les lignes du système et des substitutions. En effet, on peut facilement voir que le calcul de la matrice A^{-1} équivaut à résoudre n systèmes linéaires , ce qui s'avère bien plus coûteux que la résolution d'un seul système. Considérons à présent un système linéaire dont la matrice A est inversible et triangulaire inférieure. c'est-à-dire de la forme.

$$\begin{array}{lcccccccl}
a_{11}\,x_1 & & & & & & & = & b_1 \\
a_{21}\,x_1 & + & a_{22}\,x_2 & & & & & = & b_2 \\
\vdots & & \vdots & & \ddots & & & & \vdots \\
a_{n1}\,x_1 & + & a_{n2}\,x_2 & + & \ldots & + & a_{nn}\,x_n & = & b_n
\end{array}.$$

La matrice A étant inversible, ses termes diagonaux a_{ii}, $i = 1,\ldots..,n$, sont tous non nuls et la résolution du système est alors extrêmement simple : on calcule x_1 par une division, que l'on substitue ensuite dans la deuxième équation pour obtenir x_2, et ainsi de suite... Cette méthode, dite de « descente » (forward subtitution en anglais), s'écrit

$$\begin{aligned}
x_1 &= \frac{b_1}{a_{11}} \\
x_i &= \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}\,x_j\right), \quad i = 2,\ldots,n.
\end{aligned}$$

Algorithme 1 Algorithme de la méthode de descente (version orientée ligne)

$x_1 = b_1/a_{11}$
pour $i = 2$ à n **faire**
  $x_i = b_i$
  pour $j = 1$ à $i - 1$ **faire**
    $x_i = x_i - a_{ij}\, x_j$
  fin pour
  $x_i = x_i/a_{ii}$
fin pour

Exemple. Appliquons une approche orientée colonne pour la résolution du système

$$\begin{pmatrix} 2 & 0 & 0 \\ 1 & 5 & 0 \\ 7 & 9 & 8 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 6 \\ 2 \\ 5 \end{pmatrix}.$$

On trouve que $x_1 = 3$ et l'on considère ensuite le système à deux équations et deux inconnues pour lequel on trouve $x_2 = -1/5$

$$\begin{pmatrix} 5 & 0 \\ 9 & 8 \end{pmatrix} \begin{pmatrix} x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 2 \\ 5 \end{pmatrix} - 3 \begin{pmatrix} 1 \\ 7 \end{pmatrix}$$

Algorithme 2 Algorithme de la méthode de descente (version orientée colonne)

pour $j = 1$ à $n - 1$ **faire**
  $b_j = b_j/a_{jj}$
  pour $i = j + 1$ à n **faire**
    $b_i = b_i - a_{ij}\, b_j$
  fin pour
fin pour
$b_n = b_n/a_{nn}$

2.2 Méthode d'élimination de Gauss

Une technique de choix pour ramener la résolution d'un système linéaire quelconque à celle d'un système triangulaire et la méthode d'élimination de Gauss. Celle-ci consiste en premier lieu à transformer, par des opérations simples sur les équations, ce système en un système équivalent, c'est-à-dire ayant la(ou les) même(s) solution(s), $M\,A\boldsymbol{x} = M\,\boldsymbol{b}$, dans lequel $M\,A$ est une matrice triangulaire supérieure (on dit encore que la matrice du système est sous forme échelonnée). Cette étape de mise à zéro d'une partie des coefficients de la matrice est qualifiée d'élimination et utilise de manière essentielle le fait qu'on ne modifie pas la solution d'un système linéaire en ajoutant à une équation donnée une combinaison linéaire des autres équations. Si A est inversible, la solution du système peut ensuite être obtenue par une méthode de remontée, mais le procédé d'élimination est en fait très général, la matrice pouvant être rectangulaire.

2.2.1 Élimination de Gauss sans échange

Commençons par décrire étape par étape la méthode dans sa forme de base, dite sans échange, en considérant le système linéaire, avec A étant une matrice inversible d'ordre n. Supposons de plus que le terme a_{11} de la matrice A est non nul. Nous pouvons alors éliminer l'inconnue x_1 des lignes 2 à n du système en leur retranchant respectivement la première ligne multipliée par le coefficient a_{i1}/a_{11} , = 2,……,n.

$$a_{ij}^{(2)} = a_{ij} - \frac{a_{i1}}{a_{11}}\, a_{1j} \text{ et } b_i^{(2)} = b_i - \frac{a_{i1}}{a_{11}}\, b_1,\ i = 2, \ldots, n, j = 2, \ldots, n,$$

$$A^{(k)} = \begin{pmatrix} a_{11}^{(k)} & a_{12}^{(k)} & \ldots & \ldots & \ldots & a_{1n}^{(k)} \\ 0 & a_{22}^{(k)} & & & & a_{2n}^{(k)} \\ \vdots & \ddots & \ddots & & & \vdots \\ 0 & \ldots & 0 & a_{kk}^{(k)} & \ldots & a_{kn}^{(k)} \\ \vdots & & \vdots & \vdots & & \vdots \\ 0 & \ldots & 0 & a_{nk}^{(k)} & \ldots & a_{nn}^{(k)} \end{pmatrix}$$

le système $A^{(k)}\boldsymbol{x} = \boldsymbol{b}^{(n)}$ est triangulaire supérieur. Les quantités $a_{(kk)}^{(k)}$, k = 1,……, n - 1 sont appelées pivots et l'on a supposé qu'elles étaient non nulles à chaque étape, les formules permettant de passer du $k^{\text{ième}}$ système linéaire au $k + 1^{\text{ième}}$ se résumant à

$$a_{ij}^{(k+1)} = a_{ij}^{(k)} - \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}\, a_{kj}^{(k)} \text{ et } b_i^{(k+1)} = b_i^{(k)} - \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}\, b_k^{(k)},\ i = 2, \ldots, n, j = 2, \ldots, n.$$

Exemple d'application. Considérons la résolution par la méthode d'élimination de Gauss sans échange du système linéaire suivant

$$\begin{cases} x_1 + 2x_2 + 3x_3 + 4x_4 = 11 \\ 2x_1 + 3x_2 + 4x_3 + x_4 = 12 \\ 3x_1 + 4x_2 + x_3 + 2x_4 = 13 \\ 4x_1 + x_2 + 2x_3 + 3x_4 = 14 \end{cases}.$$

$$\begin{cases} x_1 + 2x_2 + 3x_3 + 4x_4 = 11 \\ -x_2 - 2x_3 - 7x_4 = -10 \\ -2x_2 - 8x_3 - 10x_4 = -20 \\ -7x_2 - 10x_3 - 13x_4 = -30 \end{cases}$$

$$\begin{cases} x_1 + 2x_2 + 3x_3 + 4x_4 = 11 \\ -x_2 - 2x_3 - 7x_4 = -10 \\ -4x_3 + 4x_4 = 0 \\ 4x_3 + 36x_4 = 40 \end{cases}$$

$$\begin{cases} x_1 + 2x_2 + 3x_3 + 4x_4 = 11 \\ -x_2 - 2x_3 - 7x_4 = -10 \\ -4x_3 + 4x_4 = 0 \\ 40x_4 = 40 \end{cases}.$$

Ce système triangulaire, équivalent au système d'origine, est enfin résolu par remontée :

$$\begin{cases} x_4 = 1 \\ x_3 = x_4 = 1 \\ x_2 = 10 - 2 - 7 = 1 \\ x_1 = 11 - 2 - 3 - 4 = 2 \end{cases}.$$

2.2.2 Élimination de Gauss avec échange

En considérant le cas d'une matrice A carrée inversible, nous allons maintenant décrire les modications à apporter à la méthode de Gauss sans échange pour mener l'élimination à son terme. Dans tout ce qui suit, les notations de la section 2.2.1 sont conservées. En raison de l'échange de lignes qui a éventuellement lieu avant chaque étape d'élimination, on parle de méthode d'élimination de Gauss avec échange.

Exemple d'application. Considérons la résolution du système linéaire $A\boldsymbol{x} = \boldsymbol{b}$, avec

$$A = \begin{pmatrix} 2 & 4 & -4 & 1 \\ 3 & 6 & 1 & -2 \\ -1 & 1 & 2 & 3 \\ 1 & 1 & -4 & 1 \end{pmatrix} \text{ et } \boldsymbol{b} = \begin{pmatrix} 0 \\ -7 \\ 4 \\ 2 \end{pmatrix},$$

Par application de la méthode d'élimination de Gauss avec échange.On trouve successivement

$$A^{(2)} = \begin{pmatrix} 2 & 4 & -4 & 1 \\ 0 & 0 & 7 & -\frac{7}{2} \\ 0 & 3 & 0 & \frac{7}{2} \\ 0 & -1 & -2 & \frac{1}{2} \end{pmatrix} \text{ et } \boldsymbol{b} = \begin{pmatrix} 0 \\ -7 \\ 4 \\ 2 \end{pmatrix}$$

$$A^{(3)} = \begin{pmatrix} 2 & 4 & -4 & 1 \\ 0 & 3 & 0 & \frac{7}{2} \\ 0 & 0 & 7 & -\frac{7}{2} \\ 0 & 0 & -2 & \frac{5}{3} \end{pmatrix} \text{ et } \boldsymbol{b}^{(3)} = \begin{pmatrix} 0 \\ 4 \\ -7 \\ \frac{10}{3} \end{pmatrix}$$

$$A^{(4)} = \begin{pmatrix} 2 & 4 & -4 & 1 \\ 0 & 3 & 0 & \frac{7}{2} \\ 0 & 0 & 7 & -\frac{7}{2} \\ 0 & 0 & 0 & \frac{2}{3} \end{pmatrix} \text{ et } \boldsymbol{b}^{(4)} = \begin{pmatrix} 0 \\ -7 \\ 4 \\ 4/3 \end{pmatrix},$$

On pourra remarquer que si la matrice A est non inversible, alors tous les éléments $a_{(ik)}^{(k)}$, $k \leq i \leq n$, seront nuls pour au moins une valeur de k entre 1 et n. Si $k \neq n$, on n'a dans ce cas pas besoin de réaliser l'élimination de cette $k^{\text{ième}}$ (puisque cela est déjà fait) et l'on passe simplement à l'étape suivante en posant $A^{(k+1)} = A^{(k)}$ et $\boldsymbol{b}^{(k+1)} = \boldsymbol{b}^{(k)}$. L'élimination est donc bien possible pour une matrice carrée non inversible .

2.2.5 Méthode d'élimination de Gauss_Jordan

La méthode d'élimination de Gauss-Jordan est une variante de la méthode d'élimination de Gauss ramenant toute matrice sous forme échelonnée réduite. Dans le cas d'une matrice A inversible, cette méthode revient à chercher une matrice M˜ telle que la matrice M˜ A soit non pas triangulaire supérieure mais diagonale. Pour cela, on procède comme pour l'élimination de Gauss, mais en annulant à chaque étape tous les éléments de la colonne considérée situés au dessous et au dessus de la diagonale. Si elle est bien moins efficace que la méthode d'élimination de Gauss pour la résolution de systèmes linéaires, la méthode d'élimination de Gauss_Jordan est utile pour le calcul de l'inverse d'une matrice A carrée d'ordre n.

Exemple. Soit

$$A = \begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix}$$

La matrice augmentée est

$$(A \quad I_n) = \begin{pmatrix} 2 & -1 & 0 & 1 & 0 & 0 \\ -1 & 2 & -1 & 0 & 1 & 0 \\ 0 & -1 & 2 & 0 & 0 & 1 \end{pmatrix}$$

et l'on trouve successivement

$$k = 1, \quad \begin{pmatrix} 1 & -1/2 & 0 & 1/2 & 0 & 0 \\ 0 & 3/2 & -1 & 1/2 & 1 & 0 \\ 0 & -1 & 2 & 0 & 0 & 1 \end{pmatrix}$$

$$k = 2, \quad \begin{pmatrix} 1 & 0 & -1/3 & 2/3 & 1/3 & 0 \\ 0 & 1 & -2/3 & 1/3 & 2/3 & 0 \\ 0 & 0 & 4/3 & 1/3 & 2/3 & 1 \end{pmatrix}$$

$$k = 3, \quad \begin{pmatrix} 1 & 0 & 0 & 3/4 & 1/2 & 1/4 \\ 0 & 1 & 0 & 1/2 & 1 & 1/2 \\ 0 & 0 & 1 & 1/4 & 1/2 & 3/4 \end{pmatrix}$$

$$\text{d'où } A^{-1} = \frac{1}{4} \begin{pmatrix} 3 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 3 \end{pmatrix}.$$

2.3 Interprétation matricielle de l'élimination de Gauss : la factorisation LU

La méthode de Gauss dans sa forme sans échange est équivalente à la décomposition de la matrice A sous la forme d'un produit de deux matrices, $A = LU$, avec L une matrice triangulaire inférieure (lower triangular en anglais), qui est l'inverse de la matrice M des transformations successives appliquées à la matrice A lors de l'élimination de Gauss sans échange, et U une matrice triangulaire supérieure (upper triangular en anglais), avec $U = A(n)$.

Exemple d'application. Considérons la matrice

$$A = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 10 \end{pmatrix}$$

En appliquant de l'algorithme de factorisation, on arrive à

$$A = LU = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 4 & 7 \\ 0 & -3 & -6 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\text{Si } \boldsymbol{b} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \text{ la solution de } L\boldsymbol{y} = \boldsymbol{b} \text{ est } \boldsymbol{y} = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} \text{ et celle de } U\boldsymbol{x} = \boldsymbol{y} \text{ est } \boldsymbol{x} = \frac{1}{3} \begin{pmatrix} -1 \\ 1 \\ 0 \end{pmatrix}$$

Chapitre 3

Méthodes itératives de résolution des systèmes linéaires

L'idée des méthodes itératives de résolution des systèmes linéaires est de construire une suite convergente $\left(x^{(k)}\right)_{k\in N}$ de vecteurs vérifiant.

$$\lim_{k\to+\infty} \boldsymbol{x}^{(k)} = \boldsymbol{x}$$

Où $\boldsymbol{x}$ est la solution du système (Ax=b) Dans ce chapitre, on va présenter des méthodes itératives parmi les plus simples à mettre en œuvre, à savoir les méthodes de Jacobi, de Gauss-Seidel et leurs variantes. Dans ces méthodes, qualifiées de méthodes itératives linéaires stationnaires du premier ordre, la suite $\left(x^{(k)}\right)_{k\in N}$ est obtenue, à partir d'un vecteur initial arbitraire $\boldsymbol{x}^{(0)}$, par une relation de récurrence de la forme

$$\boldsymbol{x}^{(k+1)} = B\boldsymbol{x}^{(k)} + \boldsymbol{c},\ \forall k \in \mathbb{N}$$

Où la matrice carrée B, appelée matrice d'itération de la méthode, et le vecteur $\boldsymbol{c}$ dépendent de la matrice A et du second membre $\boldsymbol{b}$ du système à résoudre.

3.1 Méthodes de Jacobi et de sur-relaxation

Observons que, si les coefficients diagonaux de la matrice A sont non nuls, il est possible d'isoler la $i^{\text{ième}}$ inconnue dans la $i^{\text{ième}}$ équation de (Ax=b) ,$1 \le i \le n$ et l'on obtient alors le système linéaire équivalent

$$x_i = \frac{1}{a_{ii}} \left(b_i - \sum_{\substack{j=1 \\ j\neq i}}^{n} a_{ij}\, x_j \right),\ i = 1, \ldots, n$$

La méthode de Jacobi se base sur ces relations pour construire, à partir d'un vecteur initial $\boldsymbol{x}^{(0)}$ donné, une suite $\left(x^{(k)}\right)_{k\in N}$ par récurrence (3.1)

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{\substack{j=1 \\ j\neq i}}^{n} a_{ij}\, x_j^{(k)} \right),\ i = 1, \ldots, n,\ k \in \mathbb{N},$$

ce qui implique que $M = D$ et $N = E + F$ dans la décomposition $A = M - N$, de la matrice A, où D est la matrice diagonale dont les coefficients sont les coefficients diagonaux de A, $d_{ij} = a_{ij}\,\delta_{ij}$, E est la matrice triangulaire inférieure de coefficients $e_{ij} = -a_{ij}$ si $i > j$ et 0 autrement, et F est la matrice triangulaire supérieure telle que $f_{ij} = -a_{ij}$ si $i < j$ et 0 autrement, avec $1 \le i; j \le n$. On a ainsi $A = D - (E + F)$ et la matrice d'itération de la méthode est donnée par $\boldsymbol{B_J = D^{-1}(E + F)}$:

On note que la matrice diagonale D doit être inversible. Cette condition n'est cependant pas très restrictive dans la mesure où l'ordre des équations et des inconnues peut être modifié. Une généralisation de la méthode de Jacobi est la méthode de sur-relaxation de Jacobi (Jacobi over-relaxation (JOR) en anglais), dans laquelle un paramètre de relaxation est introduit. Les relations de récurrence deviennent

$$x_i^{(k+1)} = \frac{\omega}{a_{ii}} \left(b_i - \sum_{\substack{i=1 \\ j \neq i}}^{n} a_{ij}\, x_j^{(k)} \right) + (1-\omega)\, x_i^{(k)}, \;\; i = 1, \ldots, n, \;\; k \in \mathbb{N}$$

Ce qui correspond à la matrice d'itération suivante

$$B_J(\omega) = \omega\, B_J + (1-\omega)\, I_n$$

Cette méthode est consistante pour toute valeur de ω non nulle et coïncide avec la méthode de Jacobi pour $\omega = 1$. L'idée de relaxer la méthode repose sur le fait que, si l'efficacié de la méthode se mesure par le rayon spectral de la matrice d'itération, alors, puisque $\rho(B_J(\omega))$ est une fonction continue de ω, on peut trouver une valeur de ω pour laquelle ce rayon spectral est le plus petit possible et qui donne donc une méthode itérative plus efficace que la méthode de Jacobi. Ce type de raisonnement s'applique également à la méthode de Gauss-Seidel .

L'étude des méthodes de relaxation pour un type de matrices donné consiste en général à déterminer, s'ils existent, un intervalle I de R ne contenant pas l'origine tel que, pour tout ω Choisi dans I, la méthode converge, et un paramètre de relaxation optimal $\omega_0 \in I$ tel que (dans le cas de la méthode de sur-relaxation)

$$\rho(B_J(\omega_0)) = \inf_{\omega \in I} \rho(B_J(\omega))$$

3.2 Méthodes de Gauss-Seidel et de sur-relaxation successive.

Remarquons à présent que, lors du calcul du vecteur $\boldsymbol{x}^{(k+1)}$ par les formules de récurrence (3.1), les premières i - 1$^{\text{ièmes}}$ composantes de $\boldsymbol{x}^{(k+1)}$ sont connues lors de la détermination de $i^{\text{ième}}$, $2 \leq i \leq n$. La méthode de Gauss-Seidel utilise ce fait en se servant des composantes du vecteur $\boldsymbol{x}^{(k+1)}$ déjà obtenues pour le calcul des suivantes. On a alors

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij}\, x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}\, x_j^{(k)} \right), \;\; i = 1, \ldots, n, \;\; k \in \mathbb{N}$$

Ce qui revient à poser $M = D - E$ et $N = F$ dans la décomposition (A=M-N), d'où la matrice d'itération associé $\boldsymbol{B_{GS} = (D - E)^{-1} F}$.

Pour que la méthode soit bien définie, il faut que la matrice D soit inversible, mais, là encore, cette condition n'est pas très restrictive en pratique.
On peut également introduire dans cette méthode un paramètre de relaxation ω. On parle alors de méthode de sur-relaxation successive (successive over-relaxation (SOR) en anglais), définie par

$$x_i^{(k+1)} = \frac{\omega}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij}\, x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}\, x_j^{(k)} \right) + (1-\omega)\, x_i^{(k)},\ i = 1, \ldots, n,\ k \in \mathbb{N}$$

et dont la matrice d'itération est

$$B_{GS}(\omega) = (I_n - \omega\, D^{-1} E)^{-1} \left((1-\omega)\, I_n + \omega\, D^{-1} F \right)$$

Cette dernière méthode est consistante pour toute valeur de ω non nulle et coïncide avec la méthode de Gauss-Seidel pour $\omega = 1$. Si $\omega > 1$, on parle de sur-relaxation, de sous-relaxation si $\omega < 1$. Il s'avère que la valeur du paramètre optimal est, en géneral, plus grande que 1, d'où le nom de la méthode.

Algorithme pour La méthode de Jacobi

$$Ax = b \iff \sum_{j=1}^{i-1} A_{ij}x_j + A_{ii}x_i + \sum_{j=i+1}^{n} A_{ij}x_j = b_i \qquad \forall\, i = 1, n$$

Fonction $x \leftarrow$ `Jacobi`(A, b, x)
Tant que (on a pas convergé) **faire**
  Pour $i = 1, n$ **faire**
    $y_i = \dfrac{b_i - \sum_{j=1}^{i-1} A_{ij}x_j - \sum_{j=i+1}^{n} A_{ij}x_j}{A_{ii}}$
  Fin Pour
  Pour $i = 1, n$ **faire**
    $x_i = y_i$
  Fin Pour
Fait

Exemple : méthode de Jacobi

$$A = \begin{pmatrix} 3 & 1 & -1 \\ 1 & 2 & 0 \\ -1 & 1 & 4 \end{pmatrix} \quad \text{avec } b = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

$$x^{(0)} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Itération 1

$$y_1 = \frac{b_1 - A_{12} * x(2) - A_{13} * x(3)}{A_{11}} = \frac{1-0+0}{3} = \frac{1}{3}$$

$$y_2 = \frac{b_2 - A_{21} * x(1) - A_{23} * x(3)}{A_{22}} = \frac{1-0-0}{2} = \frac{1}{2}$$

$$y_3 = \frac{b_3 - A_{31} * x(1) - A_{32} * x(2)}{A_{33}} = \frac{1+0-0}{4} = \frac{1}{4}$$

$$x^{(1)} = \begin{pmatrix} 1/3 \\ 1/2 \\ 1/4 \end{pmatrix}$$

Itération 2

$$y_1 = \frac{b_1 - A_{12} * x(2) - A_{13} * x(3)}{A_{11}} = \frac{1-1/2+1/4}{3} = \frac{3}{12}$$

$$y_2 = \frac{b_2 - A_{21} * x(1) - A_{23} * x(3)}{A_{22}} = \frac{1-1/3-0}{2} = \frac{1}{3}$$

$$y_3 = \frac{b_3 - A_{31} * x(1) - A_{32} * x(2)}{A_{33}} = \frac{1+1/3-1/2}{4} = \frac{5}{24}$$

$$x^{(2)} = \begin{pmatrix} 3/12 \\ 1/3 \\ 5/24 \end{pmatrix}$$

$$\text{norm}(A * x_0 - b) = 1.7321$$
$$\text{norm}(A * x_1 - b) = 0.4488$$
$$\text{norm}(A * x_2 - b) = 0.1718$$

$$[x_2 \quad A \backslash b] = \begin{pmatrix} 0.2500 & 0.2941 \\ 0.3333 & 0.3529 \\ 0.2083 & 0.2353 \end{pmatrix}$$

Les itérations de Gauss Seidel

Fonction $x \leftarrow$ Gauss_Seidel(A, b, x)

Tant que (on a pas convergé) **faire**
  Pour $i = 1, n$ **faire**
    $x_i = \dfrac{b_i - \sum_{j=1}^{i-1} A_{ij} x_j - \sum_{j=i+1}^{n} A_{ij} x_j}{A_{ii}}$;
  Fin Pour
Fait

Exemple : méthode de Gauss-Seidel

$$A = \begin{pmatrix} 3 & 1 & -1 \\ 1 & 2 & 0 \\ -1 & 1 & 4 \end{pmatrix} \quad \text{avec } b = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \qquad x^{(0)} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Itération 1

$$x_1 = \frac{b_1 - A_{12} * x(2) - A_{13} * x(3)}{A_{11}} = \frac{1-0+0}{3} = \frac{1}{3}$$
$$x_2 = \frac{b_2 - A_{21} * x(1) - A_{23} * x(3)}{A_{22}} = \frac{1-1/3-0}{2} = \frac{1}{3}$$
$$x_3 = \frac{b_3 - A_{31} * x(1) - A_{32} * x(2)}{A_{33}} = \frac{1+1/3-1/3}{4} = \frac{1}{4}$$

$$x^{(1)} = \begin{pmatrix} 1/3 \\ 1/3 \\ 1/4 \end{pmatrix}$$

Itération 2

$$x_1 = \frac{b_1 - A_{12} * x(2) - A_{13} * x(3)}{A_{11}} = \frac{1-1/3+1/4}{3} = \frac{11}{36}$$
$$x_2 = \frac{b_2 - A_{21} * x(1) - A_{23} * x(3)}{A_{22}} = \frac{1-11/36-0}{2} = \frac{25}{72}$$
$$x_3 = \frac{b_3 - A_{31} * x(1) - A_{32} * x(2)}{A_{33}} = \frac{1+11/36-25/72}{4} = \frac{69}{288}$$

$$x^{(2)} = \begin{pmatrix} 11/36 \\ 25/72 \\ 69/288 \end{pmatrix}$$

$$\text{norm}(A * x_0 - b) = 1.7321$$
$$\text{norm}(A * x_1 - b) = 0.0833$$
$$\text{norm}(A * x_2 - b) = 0.0852$$

$$[x_2 \quad A \backslash b] = \begin{pmatrix} 0.3056 & 0.2941 \\ 0.3472 & 0.3529 \\ 0.2212 & 0.2353 \end{pmatrix}$$

Les itérations de relaxation

Fonction $x \leftarrow \text{Relaxation}(A, b, x, \omega)$

Tant que (on a pas convergé) **faire**
  Pour $i = 1, n$ **faire**
    $x_i = \omega \left(\frac{b_i - \sum_{j=1}^{i-1} A_{ij} x_j - \sum_{j=i+1}^{n} A_{ij} x_j}{A_{ii}} \right) + (1 - \omega) x_i$
  Fin Pour
Fait

Pour $\omega = 1$ c'est Gauss Seidel

Quand s'arrêter ?

$$\|x^{new} - x^{old}\| < \varepsilon$$
un nombre maximal d'itérations est atteint

$$x = x^{old}$$
$$y = x^{new}$$

Chapitre 4

Calcul de valeurs et de vecteurs propres

4.1 Valeurs et vecteurs propres

Les valeurs propres d'une matrice A d'ordre n sont les n racines λ_i , $i = 1, ..., n$, réelles ou complexes, distinctes ou confondues, du polynôme caractéristique.

$$\lambda \in \mathbb{C} \rightarrow \det(A - \lambda I_n)$$

associé à A. Le spectre d'une matrice A, noté $\sigma(A)$, est l'ensemble des valeurs propres de A. On rappelle les propriétés suivantes :

$$\mathrm{tr}(A) = \sum_{i=1}^{n} \lambda_i, \ \det(A) = \prod_{i=1}^{n} \lambda_i$$

Par conséquent, la matrice A est singulière si au moins une de ses valeurs propres est nulle. Enfin, le rayon spectral d'une matrice A est le nombre défini par

$$\rho(A) = \max_{1 \leq i \leq n} |\lambda_i|$$

$\det(A^T - \lambda I_n) = \det((A - \lambda I_n)^T) = \det(A - \lambda I_n)$ d'où $\sigma(A^T) = \sigma(A)$

À toute valeur propre λ d'une matrice A est associé au moins un vecteur non nul $\boldsymbol{v}$ tel que

$$A\boldsymbol{v} = \lambda \boldsymbol{v}$$

et appelé vecteur propre de la matrice A correspondant à la valeur propre λ. Le sous-espace vectoriel constitué de la réunion de l'ensemble des vecteurs propres associés à une valeur propre λ et du vecteur nul est appelé sous-espace propre correspondant à la valeur propre λ. Il coïncide par définition avec $\mathrm{Ker}\,(A - \lambda I_n)$ et sa dimension est $n - \mathrm{rg}(A - \lambda I_n)$. On appelle cette dernière multiplicité géométrique de λ et elle ne peut jamais être supérieure à la multiplicité algébrique de λ, définie comme la multiplicité de λ en tant que racine du polynôme caractéristique. Une valeur propre ayant une multiplicité géométrique inférieure à sa multiplicité algébrique est dite défective.

4.2 Méthode de la puissance

La méthode de la puissance fournit une très bonne approximation des valeurs propres extrémales d'une matrice et de vecteurs propres associés. Dans la suite, on note λ_1 et λ_n les valeurs propres d'une matrice A d'ordre ayant ayant respectivement le plus petit et le plus grand module.

4.2.1 Approximation de la valeur propre de plus grand module

Soit A une matrice de M_n (C diagonalisable et soit V une matrice des vecteurs propres . $\boldsymbol{v}_j$, J=1,.... ,n, normalisés (c'est-à-dire de normes euclidiennes égales à 1) associés. On suppose que les valeurs propres de A sont ordonnées de la manière suivante

$$|\lambda_1| \leq |\lambda_2| \leq \cdots \leq |\lambda_n|$$

et supposons que λ_n soit de multiplicité algébrique égale à 1 et que la dernière des inégalités ci-dessus est stricte. Sous ces hypothèses, λ_n est appelée valeur propre dominante de A.
Étant donné un vecteur initial arbitraire $\boldsymbol{q}^{(0)}$ de C^n normalisé, on considère pour k = 1*;* 2..... la méthode itérative suivante

$$\begin{aligned} \boldsymbol{z}^{(k)} &= A\boldsymbol{q}^{(k-1)}, \\ \boldsymbol{q}^{(k)} &= \frac{\boldsymbol{z}^{(k)}}{\|\boldsymbol{z}^{(k)}\|_2}, \\ \nu^{(k)} &= (\boldsymbol{q}^{(k)})^* A\boldsymbol{q}^{(k)}, \end{aligned}$$

appelée méthode de la puissance.
Analysons ses propriétés de convergence. Par récurrence sur k, on peut vérifier que

$$\boldsymbol{q}^{(k)} = \frac{A^k \boldsymbol{q}^{(0)}}{\|A^k \boldsymbol{q}^{(0)}\|_2}, \ k \geq 1.$$

Cette relation rend explicite le rôle joué par les puissance de la matrice A. Ayant supposé cette dernière diagonalisable, il existe une base de vecteurs propres de A dans laquelle on peut décomposer le vecteur $\boldsymbol{q}^{(0)}$:

$$\boldsymbol{q}^{(0)} = \sum_{i=1}^{n} \alpha_i \, \boldsymbol{q}_i.$$

Comme $A\ \boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i$, on a

$$A^k \boldsymbol{q}^{(0)} = \alpha_n \lambda_n^k \left(\boldsymbol{v}_n + \sum_{i=1}^{n-1} \frac{\alpha_i}{\alpha_n} \left(\frac{\lambda_i}{\lambda_n} \right)^k \boldsymbol{v}_i \right), \ k \geq 1.$$

Puisque $\left|\frac{\lambda_i}{\lambda_n}\right|$ <1 la composante le long de $\boldsymbol{v}$1 du vecteur $A\boldsymbol{q}^{(0)}$(et donc celle de $\boldsymbol{q}^{(k)}$) augmente quand k augmente en module, tandis que les composantes suivant les autres directions diminuent. On obtient alors, en utilisant les deux dernières relations,

$$\boldsymbol{q}^{(k)} = \frac{\alpha_n \lambda_n^k (\boldsymbol{v}_n + \boldsymbol{y}^{(k)}}{\|\alpha_n \lambda_n^k (\boldsymbol{v}_n + \boldsymbol{y}^{(k)}\|_2},$$

où $\boldsymbol{y}^{(k)}$ désigne un vecteur tendant vers 0 quand k tend vers l'infini. Le vecteur $\boldsymbol{q}^{(k)}$ s'aligne donc avec un vecteur propre associé à la valeur propre dominante quand k tend vers l'infini. On a de plus l'estimation d'erreur suivante à l'étape k.

Théorème :Soit A une matrice diagonalisable d'ordre n dont les valeurs propres satisfont

$$|\lambda_1| \leq |\lambda_2| \leq \cdots < |\lambda_n|.$$

En supposant $\alpha_n \neq 0$ dans ..., il existe une constante $C > 0$ telle que

$$\|\widetilde{\boldsymbol{q}}^{(k)} - \boldsymbol{v}_n\|_2 \leq C\, |\frac{\lambda_{n-1}}{\lambda_n}|^k, \;\; k \geq 1,$$

Ou

$$\widetilde{\boldsymbol{q}}^{(k)} = \boldsymbol{v}_n + \sum_{i=1}^{n-1} \frac{\alpha_i}{\alpha_n} \left(\frac{\lambda_i}{\lambda_n}\right)^k \boldsymbol{v}_i$$

Exemple : On considère la matrice symétrique réelle A suivante

$$A = \begin{pmatrix} 1 & -3 \\ -3 & 1 \end{pmatrix}$$

On considère la méthode de la puissance itérée suivante afin de calculer la valeur propre λ_1 de plus grande valeur absolue de A : étant donné

$$q^{(0)} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

On calcule $q^{(k)} \in \mathrm{R}^2$ et $v^{(k)} \in \mathrm{R}$ pour tous $k \in \mathrm{N}$, tels que :

$$\begin{cases} q^{(k)} = \dfrac{Aq^{(k-1)}}{\|Aq^{(k-1)}\|_2}, \\ \nu^{(k)} = (q^{(k)}, Aq^{(k)}). \end{cases}$$

(1) Calculer les valeurs propres λ_1 et λ_2 de A en les ordonnant par ordre décroissant en valeur absolue ainsi que la base orthonormale de vecteurs propres associés notés $u^{(1)}$; $u^{(2)}$.

Corrigé : Le polynôme caractéristique $det(A - \lambda I) = (1-\lambda)^2 - 9$ admet pour racines les valeurs propres $\lambda_1 = 4$ et $\lambda_2 = -2$. L'espace propre associé à chaque valeur propre vérifie $Ax - \lambda_i x = 0$ soit

$$\begin{cases} (1-\lambda_i)x_1 - 3x_2 = 0, \\ -3x_1 + (1-\lambda_i)x_2 = 0. \end{cases}$$

On trouve qu'il est engendré par le vecteur propre normalisé $u^{(1)} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ -1 \end{pmatrix}$ pour la valeur propre λ_1 et par vecteur propre normalisé $u^{(2)} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ 1 \end{pmatrix}$ pour la valeur propre λ_2. On vérifie que $(u^{(1)}, u^{(2)})$ forme une base orthonormale de $\mathbb{R}^2$ car la matrice A est réelle symétrique.

(2) Calculer α_1 et α_2 tels que $q^{(0)} = \alpha_1 u^{(1)} + \alpha_2 u^{(2)}$; en déduire l'expression de $A^k \; q^{(0)}$ en fonction de $u^{(1)}$, $u^{(2)}$ et λ_1, λ_2.

Corrigé : comme la base $(u^{(1)}, u^{(2)})$ est orthonormale, on a $\alpha_1 = (q^{(0)}, u^{(1)}) = \frac{1}{\sqrt{2}}$ et $\alpha_2 = (q^{(0)}, u^{(2)}) = \frac{1}{\sqrt{2}}$. Comme $Au^{(1)} = \lambda_1 u^{(1)}$ et $Au^{(2)} = \lambda_2 u^{(2)}$, on déduit que

$$A^k q^{(0)} = \alpha_1 (\lambda_1)^k u^{(1)} + \alpha_2 (\lambda_2)^k u^{(2)}.$$

(3) Montrer par récurrence que $q^{(k)} = \frac{A^k q^{(0)}}{\|A^k q^{(0)}\|_2}$ pour tous $k \geq 1$.

Corrigé : la relation est vraie pour $k = 1$ par définition de la suite $q^{(k)}$. Supposons la vraie pour $k \geq 1$, alors on a

$$q^{(k+1)} = \frac{Aq^{(k)}}{\|Aq^{(k)}\|_2} = \frac{A}{\|Aq^{(k)}\|_2} \frac{A^k q^{(0)}}{\|A^k q^{(0)}\|_2} = \frac{\|A^k q^{(0)}\|_2}{\|A.A^k q^{(0)}\|_2} \frac{A.A^k q^{(0)}}{\|A^k q^{(0)}\|_2} = \frac{A^{k+1} q^{(0)}}{\|A^{k+1} q^{(0)}\|_2}.$$

La propriété est donc vraie pour $k+1$ et donc pour tous $k \in \mathbb{N}$.

(4) Déduire des questions 1 et 2 l'expression de $q^{(k)}$ en fonction de $u^{(1)}$, $u^{(2)}$ et λ_1, λ_2, puis calculer les limites de $q^{(k)}$ et de $v^{(k)}$ lorsque k tend vers l'infini

Corrigé : On déduit des deux questions précédentes, en utilisant l'orthonormalité de la base $(u^{(1)}, u^{(2)})$, que

$$q^{(k)} = \frac{\alpha_1 (\lambda_1)^k u^{(1)} + \alpha_2 (\lambda_2)^k u^{(2)}}{\left((\alpha_1)^2 (\lambda_1)^{2k} + (\alpha_2)^2 (\lambda_2)^{2k}\right)^{1/2}} = \frac{\alpha_1 (\lambda_1)^k}{|\alpha_1 (\lambda_1)^k|} \frac{u^{(1)} + \frac{\alpha_2}{\alpha_1}(\frac{\lambda_2}{\lambda_1})^k u^{(2)}}{\left(1 + (\frac{\alpha_2}{\alpha_1})^2 (\frac{\lambda_2}{\lambda_1})^{2k}\right)^{1/2}}.$$

Comme $\alpha_1 > 0$ et $\lambda_1 > 0$ on note que $\frac{\alpha_1 (\lambda_1)^k}{|\alpha_1 (\lambda_1)^k|} = 1$.
Comme $|\frac{\lambda_2}{\lambda_1}| = 1/2 < 1$, on déduit que

$$\lim_{k \to +\infty} q^{(k)} = u^{(1)},$$

puis

$$\lim_{k \to +\infty} v^{(k)} = (u^{(1)}, Au^{(1)}) = \lambda_1.$$

[1] Guillaume Legendre . Introduction à l'analyse numérique et au calcul scientifique (Cours *2009/2010*) université paris .